## 数据结构Huffman tree实验报告

电信八班 周宇 U201813451

1. 需求分析

编写一个程序，完成建立哈夫曼树、哈夫曼编码及译码工作，并把结果打印在屏幕上

1. 概要设计

通过哈夫曼树的建立，调用initialization（初始化），encoding（编码），decoding（译码）三个函数完成以上功能，并在主函数完成权值的输入

1. 详细设计

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

typedef struct node

{

char elem;

int rchild,lchild,parent,weight;

}hftree;

void search(hftree huff[],int j,int \*a1,int \*a2) //search 函数用于寻找最小的两个权重值

{

int i,m1,m2;

m1=1000;

m2=1000; //把m1，m2的权重赋值为最大

for(i=1;i<=j;i++)

{

if(huff[i].weight<m1&&huff[i].parent==0)

{

m2=m1; //让m2成为第二小的

m1=huff[i].weight;

\*a2=\*a1;

\*a1=i; //变换a1，a2的下标;

}

else if(huff[i].weight<m2&&huff[i].parent==0)

{

m2=huff[i].weight;

\*a2=i;

}

}

}

void encoding(hftree huff[],int n)

{

char bianma[n][10];

char cd[n];

int i,start,p,c;

cd[n-1]='\0';

for(i=1;i<=n;i++)

{

start=n-1;

c=i;

p=huff[i].parent;

while(p!=0)//huff[2n].parent=0

{

start--;

if(huff[p].lchild==c)

cd[start]='0';

else

cd[start]='1';

c=p;

p=huff[p].parent;

}

strcpy(bianma[i],&cd[start]);

}

for(i=1;i<=n;i++)

{

printf("%c",huff[i].elem);

printf("%s\n",bianma[i]);

}

}

void decoding(hftree huff[],int length,char yima[],int n) //译码从上（2n-1）开始向下

{

int temp=2\*n-1;

int i;

for(i=0;i<length;i++)

{

if(yima[i]=='0')

{

temp=huff[temp].lchild;

}

if(yima[i]=='1')

{

temp=huff[temp].rchild;

}

if(huff[temp].lchild==0)

{

printf("%c",'A'+temp-1);

temp=2\*n-1;

}

}

printf("%s",yima);

}

void initalization(hftree huff[],int w[],int n)

{

int i,a1,a2 ;//s1为权值最小的下标 s2为权值次小的下标;

for(i=1;i<=n;i++)

{

huff[i].weight=w[i];

huff[i].rchild=0;

huff[i].lchild=0;

huff[i].parent=0;

}

for(i=n+1;i<=2\*n-1;i++) //有效的节点数为2\*n-1;

{

huff[i].weight=0;

huff[i].rchild=0;

huff[i].lchild=0;

huff[i].parent=0;

}

for(i=n+1;i<=2\*n-1;i++) //注意 这里的i=2n时的huff数组未改变，在后面的编码中是循环退出的条件（2n时对应最大的那个节点）

{

search(huff,i-1,&a1,&a2);

huff[i].weight=huff[a1].weight+huff[a2].weight; //该位置权重等于最小的两个相加;

huff[i].lchild=a1;

huff[i].rchild=a2;

huff[a1].parent=i;

huff[a2].parent=i;

}

}

void main()

{

int n=6;

hftree huff[2\*n];

int i;

int w[]={5,29,7,8,14,23,3,11}; //权值的初始化；

initalization(huff,w,n);

int j=65;

for(i=1;i<=n;i++,j++)

{

huff[i].elem=j; //把j=65对应ascII码为A依次赋值给elem，即为ABCD

}

char yima[50]="11011101000100110";

int f=strlen(yima);

encoding(huff,n);

decoding(huff,f,yima,n);

}

1. 调试分析

在调试过程中对于编译码的部分较为繁琐，尤其是译码是哈夫曼树遍历后记录遍历的过程，参考了网上的代码。

1. 测试结果

A11

B0111

C010

D00

E10

F0110

ABCDEF